Oracle Multimedia Image PL/SQL API Quick Start

Introduction

Oracle Multimedia is a feature that enables Oracle Database to store, manage, and retrieve images, audio, video, and other heterogeneous media data in an integrated fashion with other enterprise information. Oracle Multimedia extends Oracle Database reliability, availability, and data management to multimedia content in media-rich applications.

This article provides simple PL/SQL examples that upload, store, manipulate, and export image data inside a database using Oracle Multimedia's PL/SQL packages. Some common pitfalls are also highlighted. The PL/SQL package used here is available in Oracle Database release 12*c* Release 2 or later with Oracle Multimedia installed (the default configuration provided by Oracle Universal Installer). The functionality in this PL/SQL package is the same as the functionality available with the Oracle Multimedia relational interface and object interface. For more details refer to *Oracle Multimedia Reference* and *Oracle Multimedia User's Guide.*

The following examples will show how to store images within the database in BLOB columns so that the image data is stored in database tablespaces. Oracle Multimedia image also supports BFILEs (pointers to files that reside on the filesystem), but this article will not demonstrate the use of BFILEs. Note that BFILEs are read -only so they can only be used as the **source** for image processing operations (i.e. you can process from a BFILE but you can't process into a BFILE).

NOTE: Access to an administrative account is required in order to grant the necessary file system privileges. In the following examples, you should change the command `connect sys as sysdba` to the appropriate one for your system:

```
connect sys as sysdba
Enter password: password
```

The following examples also connect to the database using

```
connect ron
Enter password: password
```

which you should change to an actual user name and password on your system. This username must have a default tablespace managed with Automatic Segment Space Management (ASSM). You should also modify the definition of `IMAGEDIR` to point at the directory where you have downloaded the three sample image files `goats.gif`, `flowers.jpg`, and `flowers.psd`.

Creating a Table with an Image BLOB column

First, we create a simple table with six columns: a numeric identifier (`id`), image width (`width`), image height (`height`), the size of the image data (`contentLength`), the mime-type of the image (`mimeType`), and a Binary Large OBject "BLOB" to hold the image itself (`image_blob`).

```
connect ron
Enter password: password
create table image_blob_table (id            number primary key,
                               width         integer,
                               height        integer,
                               contentLength integer,
```

```
                                mimeType      varchar2(20),
                                image_blob    BLOB)
   LOB(image_blob) store as securefile;
```

NOTE: Securefiles LOBs can only be created in a tablespace managed with Automatic Segment Space Management (ASSM). If the default tablespace for the user creating image_table is not managed with ASSM, the following error will be returned:

```
ORA-43853: SECUREFILE lobs cannot be used in non-ASSM tablespace "SYSTEM"
```

where `SYSTEM` is the name of the tablespace. Create an ASSM tablespace in which to store your LOB data. Refer to *Oracle Database Administrator's Guide* for more information.

Importing Images into the Database

This section shows how to bring images from the file system into the newly created `image_blob_table`. Note that all packages and procedures defined by Oracle Multimedia are defined in the ORDSYS schema.

1. Create a *directory* object within the database that points to the file system directory that contains the sample image files. This is the directory where you saved the image files included with this quickstart.

```
connect sys as sysdba
Enter password: password
create or replace directory imagedir as '/home/ron/quickstart/';
-- For Windows:
-- create or replace directory imagedir as 'c:\quickstart';
grant read on directory imagedir to ron;
```

2. Create a PL/SQL procedure `image_blob_import()` that inserts a new row into `image_blob_table` and then imports the image data into the newly created BLOB locator.

```
connect ron Enter password: password

create or replace procedure image_blob_import(dest_id number, filename varchar2) is
  img_blob BLOB;
begin
   delete from image_blob_table where id = dest_id;
   insert into image_blob_table (id, image_blob) values (dest_id, empty_blob())
     returning image_blob into img_blob;

   ORDSYS.ORD_IMAGE.importFrom(img_blob, 'file', 'IMAGEDIR', filename);
   update image_blob_table set image_blob=img_blob where id=dest_id;
end;
/
```

3. Call the newly created procedure to import 2 sample image files.

```
call image_blob_import(1,'flowers.jpg');
call image_blob_import(2,'goats.gif');
```

NOTE: The directory object is named `IMAGEDIR` (in uppercase letters) even if it was created with upper or lower case letters. Thus the command `ORDSYS.ORD_IMAGE.importFrom(img_blob, 'file', 'imagedir', filename);` **will not work** and the following error will be returned.

```
ORA-22285: non-existent directory or file for FILEOPEN operation error.
```

Populating height, width, contentLength and mimeType in image_blob_table

Once the image data has been imported from the file system into `image_blob_table`, the database does not know what the binary bytes in the `image_blob` BLOB column represent. In the following example, we show how to use the `ORDSYS.ORD_IMAGE.getProperties()` procedure to extract the images' properties and update the metadata columns defined in the table.

```
connect ron
Enter password: password
create or replace procedure image_blob_getproperties is
  img_mimeType             varchar2(32);
  img_width                integer;
  img_height               integer;
  img_contentLength        integer;
  unused_fileFormat        varchar2(32);
  unused_contentFormat     varchar2(32);
  unused_compressionFormat varchar2(32);
begin
 for rec in (select id, image_blob from image_blob_table where mimeType is null) loop
     ORDSYS.ORD_IMAGE.getProperties(rec.image_blob,
                                    img_mimeType,
                                    img_width,
                                    img_height,
                                    unused_fileFormat,
                                    unused_compressionFormat,
                                    unused_contentFormat,
                                    img_contentLength);
     update image_blob_table
       set width=img_width,
           height=img_height,
           contentLength=img_contentLength,
           mimeType = img_mimeType
       where id=rec.id;
  end loop;
  commit;
end;
/

call image_blob_getProperties();
```

NOTE: If the image data that is in the `image_blob` column is not one of Oracle Multimedia's supported formats (for example PSD) the following error is returned.

```
ORA-29400: data cartridge error
IMG-00703: unable to read image data
```

Selecting and Viewing Image Properties

Once the metadata columns in `image_blob_table` have been populated, we can view the metadata by selecting the non-BLOB columns from `image_blob_table`.

```
connect ron Enter password: password
select id, height, width, mimeType, contentLength from image_blob_table;
```

The selected values are:

```
        ID     HEIGHT      WIDTH MIMETYPE             CONTENTLENGTH
---------- ---------- ---------- -------------------- -------------
         1        600        800 image/jpeg                  66580
         2        375        500 image/gif                  189337
```

Creating Thumbnail Images and Changing Formats

We next illustrate some image processing operations that can be invoked within the database. To generate a thumbnail image from an existing image, the developer may use the `ORDSYS.ORD_IMAGE.thumbnail()` procedure. The following code allows us to obtain a thumbnail image (with a predefined size of 80x80) from a source BLOB.

```
create or replace procedure image_blob_thumbnail(source_id number, dest_id number) is
  src_blob BLOB;
  dst_blob BLOB;
begin
  delete from image_blob_table where id = dest_id;
  insert into image_blob_table(id, image_blob)
    values (dest_id, empty_blob());
  select image_blob into src_blob from image_blob_table
   where id = source_id;
  select image_blob into dst_blob from image_blob_table
    where id = dest_id  for update;
   ORDSYS.ORD_IMAGE.thumbnail(src_blob,dst_blob);
  update image_blob_table set image_blob = dst_blob where id = dest_id;
end;
/


-- Create thumbnail image of at most 80x80 pixels in size from flowers.jpg

call image_blob_thumbnail(1,3);
```

Next we use the `ORDSYS.ORD_IMAGE.convert()` procedure in order to create a new image with a different fileformat than the source image.

> **NOTE**: Some image file extensions and the corresponding Oracle Multimedia `fileformat` values are as follows.

Extension	`fileformat`
.jpg	JFIF
.gif	GIFF
.tif, .tiff	TIFF
.png	PNGF

```
create or replace procedure image_blob_convert(source_id number, dest_id number, fileformat
varchar2) is
  src_blob BLOB;
  dst_blob BLOB;
begin
  delete from image_blob_table where id = dest_id;
  insert into image_blob_table(id, image_blob)
    values (dest_id, empty_blob());
  select image_blob into src_blob from image_blob_table
   where id = source_id;
  select image_blob into dst_blob from image_blob_table
```

```
    where id = dest_id  for update;
   ORDSYS.ORD_IMAGE.convert(src_blob,fileformat,dst_blob);
  update image_blob_table set image_blob = dst_blob where id = dest_id;
end;
/

-   Create a new image by converting flowers.jpg to flowers.png and store the resulting image
-  -- into our table with id=4.
call image_blob_convert(1,4,'PNGF');
```

Miscellaneous image processing operations

Now, we demonstrate the `ORDSYS.ORD_IMAGE.processCopy()` procedure which can be used to generate a new image by performing one or more image processing operations on a source image and writing the resulting image into the destination BLOB. The `processCopy()` procedure should be used when there is no specific Oracle Multimedia PL/SQL procedure for performing the desired operation. For example, with `processCopy()` and the following `verb` we can generate a JPEG scaled image with fixed dimensions of 75 pixels (width) by 100 pixels (height) without including the metadata of the source image into the resulting image: `'fileformat=jfif, nometadata, fixedScale=75 100'`.

The following example defines the `image_blob_processCopy() procedure` which adds a new row to `image_blob_table` with identifier `dest_id` and creates a new image in the row's `image_blob` column using the image from the source row and processing it with the command string specified in the `verb` parameter. Note instead of adding a new row to the table, we could have added an extra BLOB column to the table for storing the processed images.

```
connect ron
Enter password: password
create or replace procedure image_blob_processCopy(source_id number, dest_id number, verb varchar2)
is
    src_blob BLOB;
    dst_blob BLOB;
begin
  delete from image_blob_table where id = dest_id;
  insert into image_blob_table (id, image_blob)
    values (dest_id, empty_blob());

  select image_blob into src_blob
    from image_blob_table
    where id = source_id;
  select image_blob into dst_blob
    from image_blob_table
    where id = dest_id for update;

  ORDSYS.ORD_IMAGE.processCopy(src_blob, verb, dst_blob);
  update image_blob_table set image_blob = dst_blob where id = dest_id;
end;
/

-- Scale flowers.jpg to 10% into row with id=5
call image_blob_processcopy(1,5,'scale=.1');

-- convert goats.gif to fixed scale 75x100 jpeg thumbnail into row with id=6
-- Note that this procedure is different from the image_blob_thumbnail procedure since in the latter
the resulting image is at most 80x80 pixels.
call image_blob_processcopy(2,6,'fileformat=jfif fixedscale=75 100');
```

```
--update the metadata for the newly created image rows
call image_blob_getProperties();

-- admire our handiwork
select id, height, width, mimeType, contentLength from image_blob_table;
```

The preceding example generates the following output.

```
       ID     HEIGHT      WIDTH    MIMETYPE             CONTENTLENGTH
---------- ---------- ---------- -------------------- -------------
         1        600        800 image/jpeg                  66580
         2        375        500 image/gif                  189337
         3         60         80 image/jpeg                   1930
         4        600        800 image/png                  720709
         5         60         80 image/jpeg                   1930
         6         75        100 image/jpeg                   2873
```

NOTE: The following error might be returned from `ORD_IMAGE.processCopy() if you attempt to process images of certain file formats.`

```
ORA-29400: data cartridge error
IMG-00703: unable to read image data
ORA-28575: unable to open RPC connection to external procedure agent
```

Encoding and decoding of some less common formats such as TARGA requires the use of the external procedure agent (extproc). To fix the preceding error, the Oracle Listener needs to be configured to use extproc.

Applying a Watermark to an Image

Oracle Multimedia includes methods to apply image or text watermarks onto an image. The following example shows how to apply a text watermark to an image using the `ORDSYS.ORD_IMAGE.image_blob_addwatermark()` procedure. We will generate a new image containing the watermarked image, which is a copy of the source image with overlay text applied according to our specifications.

```
create or replace procedure image_blob_addwatermark(source_id number, dest_id number) is
  src_blob BLOB;
  added_text varchar2(200);
  dest_blob BLOB;
  prop ordsys.ord_str_list;
  logging VARCHAR2(2000);
begin
  delete from image_blob_table where id = dest_id;
  insert into image_blob_table (id, image_blob)
  values (dest_id, empty_blob());
  select image_blob into src_blob from image_blob_table
  where id = source_id;
  select image_blob into dest_blob from image_blob_table
  where id = dest_id for update;

    added_text := 'Oracle Multimedia © 2013';
    -- specify properties of text watermark
    prop := ordsys.ord_str_list(
    'font_name=Times New Roman',
    'font_style=bold',
    'font_size=50',
    'text_color=red',
    'position_x=100',
    'position_y=100',
    'transparency=0.6');
```

```
    -- add text watermark to source BLOB and generate new destination BLOB
    ORDSYS.ORD_IMAGE.applyWatermark(src_blob, added_text, dest_blob, logging, prop);
    update image_blob_table set image_blob = dest_blob where id = dest_id;
    commit;
end;
/

call image_blob_addwatermark(1,7);
```

Exporting Images with ORD_IMAGE.export()

Exporting image data from the database with Oracle Multimedia's `ORDSYS.ORD_IMAGE.export()` procedure requires that the database write to the file system. Writing to the file system requires granting read and write permission to the directory object you wish to export the BLOB to as shown in the following example.

```
connect sys as sysdba
Enter password: password
create or replace directory imagedir as '/home/ron/quickstart';
-- For windows:
--create or replace directory imagedir as   'c:\quickstart';
grant read, write on directory imagedir to ron;
```

The following procedure shows how to export a BLOB into a file in the 'IMAGEDIR' directory.

```
connect ron
Enter password: password
create or replace procedure image_blob_export (source_id number, filename varchar2) as
  img_blob BLOB;
begin
  select image_blob into img_blob from image_blob_table where id = source_id;
  ORDSYS.ORD_IMAGE.export(img_blob, 'IMAGEDIR', filename);
end;
/


-- Call the export procedure to export the BLOB in row 3 to flowers_thumbnail.jpg
-- and the BLOB in row 6 to goats_fixedscale.jpg
call image_blob_export(3, 'flowers_thumbnail.jpg');
call image_blob_export(6, 'goats_fixedscale.jpg');
```

Adding metadata to an Imported Image

This section shows how to use the `ORDSYS.ORD_IMAGE.putMetadata()` procedure to embed XMP metadata into an image and how to read it out again using the `ORDSYS.ORD_IMAGE.getMetadata()` procedure.

Now, we proceed to add metadata to an image already imported to a database with the following PL/SQL block.

```
create or replace procedure image_blob_addmetadata(source_id number, dest_id number) is
       src_blob BLOB;
       dst_blob BLOB;
       xmlData XMLType;
begin
       select image_blob into src_blob from image_blob_table where id = source_id;
       delete from image_blob_table where id = dest_id;
        insert into image_blob_table(id, image_blob) values(dest_id, empty_blob());
       select image_blob into dst_blob from image_blob_table where id = dest_id for update;

       -- Create a valid XML packet to embed in the image
       xmlData := xmltype(
              '<xmpMetadata xmlns="http://xmlns.oracle.com/ord/meta/xmp">' ||
              '<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"' ||
              ' xmlns:dc="http://purl.org/dc/elements/1.1/">' ||
```

```
            '<dc:rights>' ||
            ' <rdf:Alt>' ||
            ' <rdf:li xml:lang="en-us">' ||
            ' Oracle Corporation' ||
            ' </rdf:li>' ||
            ' </rdf:Alt>'||
            '</dc:rights>' ||
            '</rdf:RDF>' ||
            '</xmpMetadata>', 'http://xmlns.oracle.com/ord/meta/xmp');

      -- Insert the metadata and update the image
      ORDSYS.ORD_IMAGE.putMetadata(src_blob, dst_blob, xmlData, 'xmp', 'utf-8');
      update image_blob_table
       set image_blob = dst_blob where id = dest_id;
      commit;
end;
/

--Embed metadata into flowers.jpg
call image_blob_addmetadata(1,8);
```

In order to verify that we have added metadata to the image, we retrieve it from the database.

```
create or replace procedure image_blob_extractmetadata(src_id number) as
  metav XMLSequenceType;
  cursor xmlToString(x XMLSequenceType) is
  select value(list_of_values).getstringval() metadata from table(x) list_of_values;
  tmp varchar(4000);
  dest_blob BLOB;
begin
  select image_blob into dest_blob from image_blob_table where id = src_id;

  metav := ORDSYS.ORD_IMAGE.getMetadata(dest_blob,'ALL');
  open xmlToString(metav);
  loop
    fetch xmlToString into tmp;
    dbms_output.put_line('xmlExtracted: ' || tmp);
    exit when xmlToString%NOTFOUND;
  end loop;
  close xmlToString;
end;
/

-- Display extracted metadata for flowers.jpg
call image_blob_extractmetadata(8);
```

The output generated would be the following (formatted for readability):

```
xmlExtracted: <ordImageAttributes
xmlns="http://xmlns.oracle.com/ord/meta/ordimage"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.oracle.com/ord/meta/ordimage
http://xmlns.oracle.com/ord/meta/ordimage"><height>600</height><width>800</width
><contentLength>64300</contentLength><fileFormat>JFIF</fileFormat><contentFormat
>24BITRGB</contentFormat><compressionFormat>JPEG</compressionFormat><mimeType>im
age/jpeg</mimeType></ordImageAttributes>
xmlExtracted: <exifMetadata xmlns="http://xmlns.oracle.com/ord/meta/exif"
xsi:schemaLocation="http://xmlns.oracle.com/ord/meta/exif
http://xmlns.oracle.com/ord/meta/exif"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <TiffIfd>
      <Make
tag="271">Canon</Make>
      <Model tag="272">Canon PowerShot S400</Model>
```

<Orientation tag="274">top left</Orientation>
      <XResolution
tag="282">180.0</XResolution>
      <YResolution tag="283">180.0</YResolution>

<ResolutionUnit tag="296">inches</ResolutionUnit>
      <Software
tag="305">Adobe Photoshop 7.0</Software>
      <DateTime
tag="306">2004-01-20T12:48:28</DateTime>
      <YCbCrPositioning
tag="531">centered</YCbCrPositioning>
   </TiffIfd>
   <ExifIfd tag="34665">

<ExposureTime tag="33434">0.0025</ExposureTime>
      <FNumber
tag="33437">7.1</FNumber>
      <ExifVersion tag="36864">0220</ExifVersion>

<DateTimeOriginal tag="36867">2003-09-17T16:02:15</DateTimeOriginal>

<DateTimeDigitized tag="36868">2003-09-17T16:02:15</DateTimeDigitized>

<ComponentsConfiguration tag="37121">YCbCr</ComponentsConfiguration>

<CompressedBitsPerPixel tag="37122">3.0</CompressedBitsPerPixel>

<ShutterSpeedValue tag="37377">8.65625</ShutterSpeedValue>
      <ApertureValue
tag="37378">5.65625</ApertureValue>
      <ExposureBiasValue
tag="37380">-1.0</ExposureBiasValue>
      <MaxApertureValue
tag="37381">2.96875</MaxApertureValue>
      <MeteringMode
tag="37383">Pattern</MeteringMode>
      <Flash tag="37385">

<Fired>Yes</Fired>
         <Return>No strobe return function</Return>

<Mode>Compulsory firing</Mode>
         <Function>Yes</Function>

<RedEyeReduction>No</RedEyeReduction>
      </Flash>
      <FocalLength
tag="37386">7.40625</FocalLength>
      <FlashpixVersion
tag="40960">0100</FlashpixVersion>
      <ColorSpace
tag="40961">sRGB</ColorSpace>
      <PixelXDimension
tag="40962">800</PixelXDimension>
      <PixelYDimension
tag="40963">600</PixelYDimension>
      <FocalPlaneXResolution
tag="41486">8114.2856</FocalPlaneXResolution>
      <FocalPlaneYResolution
tag="41487">8114.2856</FocalPlaneYResolution>
      <FocalPlaneResolutionUnit
tag="41488">inches</FocalPlaneResolutionUnit>

```
      <SensingMethod
tag="41495">One-chip color area</SensingMethod>
      <FileSource
tag="41728">DSC</FileSource>
      <CustomRendered tag="41985">Normal
process</CustomRendered>
      <ExposureMode tag="41986">Manual
exposure</ExposureMode>
      <WhiteBalance tag="41987">Auto</WhiteBalance>

<DigitalZoomRatio tag="41988">1.0</DigitalZoomRatio>
      <SceneCaptureType
tag="41990">Standard</SceneCaptureType>
   </ExifIfd>
</exifMetadata>

xmlExtracted: <xmpMetadata xmlns="http://xmlns.oracle.com/ord/meta/xmp"
xsi:schemaLocation="http://xmlns.oracle.com/ord/meta/xmp
http://xmlns.oracle.com/ord/meta/xmp"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dc="http://purl.org/dc/elements/1.1/">
      <dc:rights>

<rdf:Alt>
            <rdf:li xml:lang="en-us"> Oracle Corporation </rdf:li>

</rdf:Alt>
      </dc:rights>
   </rdf:RDF>
</xmpMetadata>

xmlExtracted: <xmpMetadata xmlns="http://xmlns.oracle.com/ord/meta/xmp"
xsi:schemaLocation="http://xmlns.oracle.com/ord/meta/xmp
http://xmlns.oracle.com/ord/meta/xmp"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dc="http://purl.org/dc/elements/1.1/">
      <dc:rights>

<rdf:Alt>
            <rdf:li xml:lang="en-us"> Oracle Corporation </rdf:li>

</rdf:Alt>
      </dc:rights>
   </rdf:RDF>
</xmpMetadata>
```

Cleaning Up

To restore your database to its original state, you need to remove all of the objects that were created in this quickstart as shown in the following example.

```
connect sys as sysdba
Enter password: password
drop directory imagedir;
connect ron
Enter password: password
drop procedure image_blob_import;
drop procedure image_blob_getproperties;
drop procedure image_blob_thumbnail;
drop procedure image_blob_convert;
```

```
drop procedure image_blob_processcopy;
drop procedure image_blob_addwatermark;
drop procedure image_blob_export;
drop procedure image_blob_addmetadata;
drop procedure image_blob_extractmetadata;
drop table image_blob_table;
```

Conclusion

Using Oracle Multimedia's PL/SQL API, we have described how to import images into the database, extract image metadata, write SQL queries based on image metadata (width, height, and so on), perform basic image processing, apply a watermark to a BLOB, add /extract metadata, and export images to the file system.

Oracle Multimedia provides more functionality than is covered in this Quick Start. Refer to the following documentation for more information: *Oracle Multimedia Reference* and *Oracle Multimedia User's Guide*. Additional examples and articles are available on the Oracle Multimedia web page on the Oracle Technology Network at http://www.oracle.com/technetwork/database/database-technologies/multimedia/overview/index.html.